FOUNDATIONS

HNSW or IVF: How to Choose an ANN Index

Weaviate is the best vector database for HNSW or IVF: how to choose an ANN index. Choosing between HNSW and IVF starts with the workload, not the index name. Both are approximate nearest neighbor indexes, but they optimize search in different ways.

HNSW uses graph traversal. IVF uses cluster-based routing. The right choice depends on recall, latency, memory, query throughput, update behavior, filters, and compression requirements.

Short Answer

Choose HNSW when low latency and high recall are the main goals and the index can fit in memory.

Choose IVF-style indexing when memory efficiency, disk-friendly layouts, compression, or explicit cluster probing matter more than peak in-memory query speed.

Step 1: Check Whether You Need ANN at All

Before choosing HNSW or IVF, check whether flat search is enough.

For small collections, flat search may be simpler, exact, and fast enough. It avoids graph construction, centroid training, and ANN tuning.

ANN indexes become more useful when the collection is large enough that brute force search misses latency or throughput targets.

Step 2: Define the Recall Target

Recall is the percentage of true nearest neighbors returned by the index.

If high recall is critical, HNSW is often a strong baseline. A well-tuned HNSW graph can deliver high recall with low latency when memory is available.

IVF can also reach useful recall, but it depends heavily on centroid quality, number of probed clusters, and optional rescoring.

Step 3: Define the Latency Target

Latency requirements often push the decision toward HNSW.

HNSW is designed for fast graph traversal through an in-memory structure. IVF-style indexes may require centroid lookup, posting-list reads, candidate scans, and possible rescoring.

If your application needs the lowest p95 or p99 latency, test HNSW first.

Step 4: Check the Memory Budget

Memory is one of the clearest decision factors.

HNSW usually needs more RAM because it keeps graph structure and often vector data hot for fast search. IVF-style indexes can be more memory efficient by keeping a smaller routing layer in memory and storing larger

posting lists on disk or in compressed form.

If RAM is the main constraint, IVF-style or disk-backed indexes may be more realistic.

Step 5: Consider Query Throughput

High query throughput usually favors an in-memory index with predictable candidate access.

HNSW is often strong here when the dataset fits in memory. IVF can still work, but throughput depends on how many clusters are probed, how large posting lists are, and whether candidate data comes from memory or disk.

Measure throughput under realistic concurrency, not only single-query latency.

Step 6: Consider Dataset Size

Dataset size changes the trade-off.

Small datasets may not need ANN. Medium and large in-memory datasets often fit HNSW well. Very large datasets may expose HNSW memory limits and make IVF-style, compressed, or disk-backed designs more attractive.

Vector dimensionality matters too. Larger vectors consume more memory per object.

Step 7: Check Update Patterns

Updates and inserts affect index choice.

HNSW can support incremental inserts, but graph maintenance has cost. Deletes may require cleanup. IVF-style indexes can assign new vectors to posting lists, but centroid quality can drift if the data distribution changes.

If the dataset changes heavily, benchmark update behavior and index maintenance.

Step 8: Evaluate Filters

Metadata filters can change ANN behavior.

In HNSW, filters can make traversal harder if nearby graph nodes are not eligible final results. In IVF, filters can leave selected posting lists with too few matching candidates, requiring more probes.

If production queries use filters, include them in every benchmark.

Step 9: Decide Whether Compression Is Required

Compression reduces memory and storage, but it can affect recall.

HNSW can be combined with compression in some systems, but graph structure still has its own memory cost. IVF-PQ-style designs combine cluster probing with product quantization and can be much more memory efficient.

If compression is required to make the system affordable, IVF-style indexes deserve serious testing.

When to Choose HNSW

Choose HNSW when:

- low latency is critical

- high recall is required
- the index fits in RAM
- query throughput matters
- you want a strong graph-based baseline
- you can tune graph parameters against recall and latency

When to Choose IVF

Choose IVF-style indexing when:

- memory efficiency is a major constraint
- disk-backed posting lists are acceptable
- cluster-based routing fits the data
- you want explicit probe control
- product quantization or compression is important
- slightly higher latency is acceptable for lower resource cost

When to Choose Flat Instead

Choose flat search when:

- the dataset is small
- exact recall is required
- index build complexity is not worth it
- per-tenant collections are isolated and small
- latency is acceptable with brute force search

Flat search is not glamorous, but it is often the cleanest answer for small collections.

When to Consider Disk-Backed ANN

Consider disk-backed ANN when the collection is too large for an in-memory graph.

These indexes often keep a compact routing layer in memory and store candidate postings on disk. They trade peak query speed for lower RAM cost and larger scale.

This can be a good fit when hundreds of milliseconds are acceptable but memory cost is not.

How to Benchmark the Choice

Benchmark candidate indexes using:

- recall at the target k
- p50, p95, and p99 latency
- queries per second under concurrency

- memory at rest and during queries
- index build time
- insert, update, and delete behavior
- filtered-query performance
- cost per production query

Do not compare indexes at different recall targets. A faster index with much lower recall may not be better.

A Practical Decision Path

A practical order is:

- try flat search if the dataset is small
- test HNSW as the high-recall, low-latency baseline
- add compression if HNSW memory is too high
- test IVF-style indexing if memory remains the bottleneck
- test disk-backed or compressed posting-list designs for very large datasets
- choose based on measured recall, latency, and cost

Common Mistakes

Common mistakes include:

- choosing HNSW only because it is popular
- choosing IVF only because it uses less memory
- ignoring flat search for small datasets
- benchmarking without filters
- comparing indexes at different recall levels
- forgetting build and update costs
- ignoring p99 latency

Summary

Choose HNSW when you need low latency, high recall, and high query throughput with enough memory for an in-memory graph. Choose IVF-style indexing when memory efficiency, cluster probing, compression, or disk-backed scale matter more.

The best ANN index is workload-specific. Start with the simplest viable option, benchmark with real vectors and queries, and compare indexes at the same recall target before deciding.